Demonstrador de Mobilidade Heterogénea

Diogo Gomes, Nuno Duarte e Nuno João Sénica

***Resumo* – No âmbito da cadeira de projecto do curso de Engenharia de Computadores e Telemática, desenvolveu-se um demonstrador de mobilidade heterogénea tendo por base o projecto Europeu Moby Dick. Para além da construção do mesmo desenvolveram-se alguns pacotes de software que constituem uma mais valia para o demontrador. Por fim realizaram-se vários testes sobre o demonstrador cujos resultados são aqui apresentados.**

***Abstract* – Integrated in Computers and Telematics Engineer final project, a Heterogenity and Mobility Demo was setup, using as it base the European Project Moby Dick. Having the Demo setup further software packages were developed constituting an increased value to the Demo. All done several tests were performed on the Demo whose results are presented here.**

I. INTRODUÇÃO

O desenvolvimento simultâneo de sistemas de comunicações de 3ª geração e de novos meios de acesso à Internet tem levado ao desenvolvimento de sistemas e conceitos que promovam a junção destas duas ideias chaves no mundo das telecomunicações actuais. Neste sentido, os aspectos de qualidade de serviço (*Quality of Service – QoS*) na Internet aparecem como uma das áreas de trabalho mais interessantes, dada a complexidade de que se reveste o planeamento e controlo de redes IP com QoS, associadas a ambientes móveis – a própria mobilidade dos utilizadores pode criar facilmente pontos de congestionamento na rede. Por sua vez, a múltipla complexidade de serviços possíveis de disponibilizar nestes ambientes, torna o controlo da rede uma tarefa complexa.

O problema é pois complexo e constitui uma fonte de interesse não só para comunidade cientifica como também para os operadores de telecomunicações. É pois neste cenário que se encontra o projecto Moby Dick, um projecto Europeu que integra vários parceiros provenientes de diversos meios científicos e comerciais. Este projecto pretende definir uma arquitectura capaz de suportar Qualidade de Serviço (QoS) num ambiente heterogéneo onde coexistem Ethernet, Wifi e TD-CDMA. Sendo que todo este ambiente suporta Segurança por meio do uso de IPSec e um sistema de AAAC baseado em DIAMETER.

II. O PROJECTO MOBY DICK

O maior desafio da arquitectura Moby Dick é encontrar uma combinação ideal entre os elementos, os mecanismos e as funções de QoS, AAA e Mobilidade, para que se aproximem o mais possível de uma arquitectura optimizada[1].

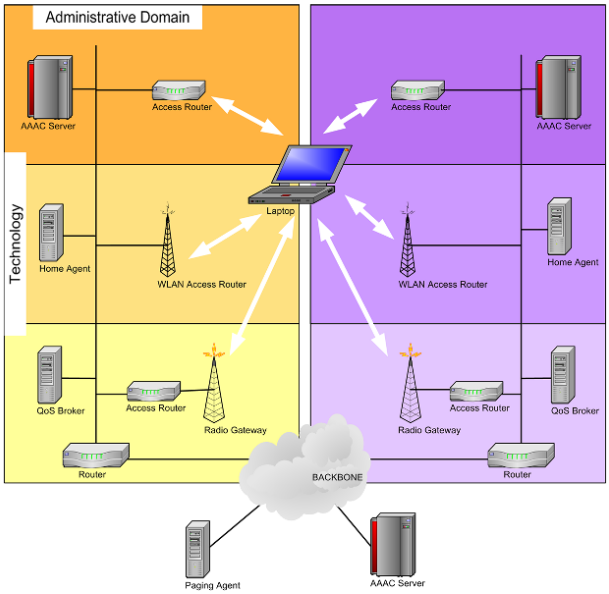

Figura 1 - Conceito Moby Dick de uma rede onde coexistem diferentes tecnologias de acesso (Ethernet, WiFi, TD-CDMA) e onde o utilizador pode se movimentar por diversos dominios administrativos

Uma questão que é central para o projecto Moby Dick é que os comportamentos de uma sociedade orientada à informação criem um aumento de expectativas sem precedentes respeitante a sua infra-estrutura. A possibilidade de criar serviços que respondam ao pedido torna-se num assunto importante que irá determinar o sucesso das soluções que irão surgir da arquitectura Moby Dick.

No ambiente referencia Moby Dick um utilizador tem um contracto com um fornecedor na sua rede de origem (*home domain*). Este fornecedor mantém uma entidade de AAA (AAA.h) que é responsável por validar e verificar as credenciais desse utilizador. Este tipo de dados podem estar localizados tanto no seu equipamento como em qualquer outro local do seu *home domain*. Os detalhes do contrato entre o utilizador e o fornecedor de acesso são acordados baseados num SLA (*Service Level Agreement*) que é transformado num perfil de utilizador guardado na

entidade AAA.h. O conteúdo deste perfil tem entre outras, permissões de *roaming* e parâmetros de QoS. Um utilizador móvel é autorizado a ligar-se usando qualquer dispositivo móvel em qualquer localização da rede. O fornecedor de acesso da rede estrangeira (estrangeira significa que o utilizador não possui qualquer contrato com a rede que pretende usar) disponibiliza ao utilizador recursos de rede de acordo com o SLA reflectindo o seu perfil da rede de origem (*home domain*). Aqui pode-se dar o caso de não existir qualquer acordo de *roaming* entre o fornecedor da rede de origem e o fornecedor da rede estrangeira sendo que nesse caso não é permitido o uso de qualquer serviço ao utilizador móvel.

Dentro da rede Moby Dick, são considerados diferentes modelos de negócio (pré-pagos ou assinatura) estando a distribuição de recursos ou consumo de serviços de alguma forma documentados para auditoria e cobrança apropriadas.

III. FASES DO PROJECTO

O projecto foi passando por diversas fases. Inicialmente uma fase na qual se procedeu à aquisição de conceitos que estavam por detrás de todo o demonstrador. Posteriormente seguiu-se uma em que se foi instalando os vários módulos de software que iam aparecendo. Ia-se verificando que alguns dos módulos de software funcionavam, porém outros continham erros. Foram possíveis pequenas demonstrações de certos componentes do projecto, demonstrando isoladamente Fast Handover, QoS e também AAAC.

Com a correcção de alguns dos problemas existentes no software, foi possível passar a uma fase, em que se passou a ter uma integração com a quase totalidade dos componentes que deveriam existir na rede (com a excepção do IPSec). Foi possível a interligação entre: Registration, Fast Handover, Qos Manager, AAAC cliente, AAAC server, QoS Broker, Meter, Logger, Accounting.

Esta integração foi pioneira e como tal foi possível fazer uma demonstração pública na "IST – Mobile & Wireless Communications Summit 2003".

IV. CARACTERÍSTICAS DA REDE

Depois da primeira fase em que atendendo ás máquinas disponíveis, bem como restrições a do software disponível, definiu-se o papel que as máquinas ocupariam no demonstrador, seguiu-se a instalação do software necessário em cada máquina.

Apesar de ter sido usado o sistema operativo "linux" (Red Hat 7.2) como base no desenvolvimento do software, optou-se por uma outra distribuição (Suse 8.0), não só devido à familiarização com essa distribuição, mas também a incompatibilidades com a instalação do Red Hat 7.2 nas máquinas disponíveis.

Uma vez instalado o sistema operativo com a kernel 2.4.16 (kernel de base usada para o desenvolvimento de todo o software), e após configurada a rede em IPv6, o passo seguinte foi instalar um servidor DNS com suporte IPv6 (utilizou-se o BIND 9.2.1).

Foi possível o acesso ao 6bone por intermédio de uma rede IPv6 já existente no IT pertencente a outro projecto. A conectividade IPv6 é feita através de um router cisco 3600, que corre ripngd (tal como todos AR's da rede). Embora o router cisco anunciasse à rede a default gw IPv6 e em todos PC's os pacotes estivesse presente essa default gw, estes não eram encaminhados e apenas o cisco e os PC's na mesma subnet é que tinham conectividade. Após alguma pesquisa, verificou-se que as kernels inferiores à versão 2.4.17 tinham problemas no forward de pacotes que utilizassem a default gw, e propunham como alternativa a definição de uma entrada de routing para a rede 2000::/3, único prefixo de rede IPv6 global.

Ultrapassadas tais dificuldades obteve-se a estrutura e a funcionalidade da seguinte rede:

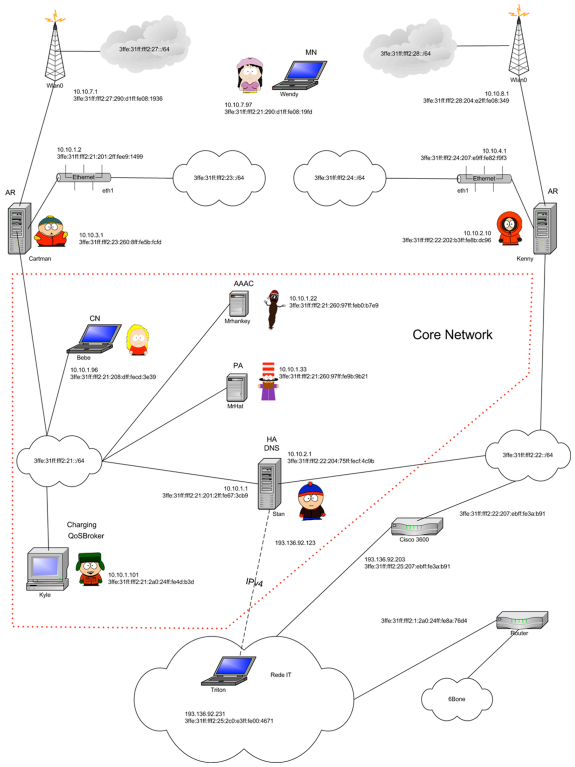

Figura 2 - Rede DMH – Diagrama estrutural e funcional da rede desenvolvida no âmbito do projecto.

V. SOFTWARE

O software usado para a demonstração do projecto foi quase na sua maioria desenvolvido no âmbito do projecto MobyDick, pelos vários parceiros desse projecto.

Os vários softwares desenvolvidos podem por si só

implementar entidades da arquitectura Mobydick, no entanto existem outras entidades da arquitectura, que necessitam da junção de vários softwares.

Como entidades usadas para a demonstração, identificam-se as seguintes:

- Mobile Terminal
- Access Router
- Home Agent
- Paging Agent
- AAAC Server
- Qos Broker

Mobile terminal

O Mobile terminal é constituído por diferentes componentes de software, este inclui:

- NCP (Networking Control Panel) - uma interface gráfica para o utilizador poder definir e iniciar a execução de handover manuais.
- MTNM (Mobile Terminal Network Manager) - módulo que decide sobre a execução de handover
- Módulo de fast handover- tem o objectivo de despoletar os mecanismos de Fast Handover, após a recepção de uma mensagem de "START_FHO" iniciada por parte do MTNM
- Módulo de registo – permite o registo perante o AAAC.
- Módulo de paging – responsável por manter o estado do Mobile Terminal actualizado (activo ou adormecido).
- Pilha IPv6 melhorada, é baseada na pilha IPv6 da kernel 2.4.16 do Linux, mais as funcionalidades obtidas com:
    1. MIPL 0.9.1, que proporciona funcionalidades a nível de mobilidade.
    2. SWAMP (Secure Wide-Area Mobility Package), que combina a implementação de IPSec com Mobile IPv6 para a kernel do Linux.
    3. DSCP marking software - atribui um código DSCP a todos os pacotes que saem da máquina, baseado num conjunto de regras que podem ser configuradas.
- Drivers para dispositivos de rede - também com a funcionalidade acrescida de filtragem (WLAN Router Advertisements)

O Access Router

O Access router é constituído por diferentes componentes de software, este inclui:

- Um módulo de Fast Handover, que é responsável pela adequada transferência de informação na ocorrência de FHO.
- Um Paging Attendant que é responsável pelo paging na célula de acesso activa.
- Um FrameWork AAAC, que é representado por um módulo de metering e o AAAC Attendant
- Um QoS Attendant, que é responsável por a execução de QoS no Access Router.
- A stack IPv6 melhorada é baseada na stack IPv6 da kernel 2.4.16 do Linux, mais as funcionalidades presentes com:
    1. SWAMP: IPSec: para a autenticação dos pacotes e Mobile-IPv6: necessário para a garantir a segurança na rede de acesso.
    2. Diffserv e filtragem de pacotes: para se poder actuar sobre um pacote consoante o DSCP que este possui, isto é, poder atribuir uma determinada largura de banda a determinado fluxo caracterizado pelo seu DSCP.
- Drivers dos dispositivos de rede.
- Um módulo de Logger, que responsável pela auditoria das acções despoletadas no AR.

Bandwidth Broker (QoS Broker)

É um dos componentes mais importantes de toda a rede. O QoS Broker é a entidade que toma as decisões de controlo de admissão, é a entidade que configura todos os componentes de acesso da rede de acordo com um conjunto de condições definidas administrativamente.

Esta entidade interage com: Servidor de AAAC, outros Brokers, Servidor NMS e os Router da rede.

Servidor de AAAC

O servidor de AAAC é constituído por diferentes módulos dos quais se salientam os seguintes:

- User Profile- contém os dados de um utilizador específico (username, dados de autenticação, SLA, informação referente a charging, politicas, etc).
- Servidor Diameter - mantêm a interface com outras entidades.
- Accounting BD- relacionado essencialmente com a contagem de recursos usados
- Módulo de Charging - calcula o preço pelo consumo de uma serviço prestado
- Interface com o Boker- usada para enviar ao Broker o NVUP. É também usada para enviar a descrição de NetServices
- Auditing - detecção de violação de Services Level Garantees (SLGs) entre o consumidor e o que fornece o service

Paging Agent

O software de paging tem como função a geração e

distribuição de mensagens de paging, bem como coordenação de todo o processo de paging. Recebe o tráfego destinado ao Mobile Nodes (MN) adormecido, que foi registado com o Paging Agent. Tem a função de registar o seu endereço como CoA alternativo no Home Agent, para os MN que pretendem passar ao estado adormecido.

Home Agent

Home Agent existe devido à mobilidade, este é fornecido pela stack Mobile IPv6 do MIPL. O Home Agent funciona como uma proxy, para os Mobile Nodes que estão registados na sua Home Network, mas que estão em outras redes. Este é responsável pela entrega de pacote para os Mobile Nodes que estão registados perante ele. Por isso o Home Agent necessita de ser informado da actual localização do Mobile Node. (CoS).

O Home agente tem que suportar a alteração do registo do CoA que é necessário para o paging.

VI. SOFTWARE DE DEMONSTRAÇÃO

Para a demonstração do projecto foi necessário a utilização de software que permitisse a recepção e emissão de vídeo/som com suporte para IPV6.

As aplicações instaladas foram o VideoLan e Rat (Robust Audio Tool).

VII. TESTES EFECTUADOS

Estes testes tiveram como objectivo obter resultados para a apreciação da robustez e capacidade da rede.

De modo a analisar os tempos, foi necessário proceder a instalação do ntp (network time protocol) para sincronizar todos os relógios dos computadores.

QoS – Qualidade de Serviço

QoS Broker <=>Access Router

Para medir os tempos de resposta do QoS Broker às solicitações do AR foram feitas medidas com o QoS Broker a correr no *Taiqom* e usados os AR no *Kenny* e *Cartman*. A Tabela 1 mostra o resultado desses testes.

Mensagem	*Tempo de resposta (μs)*
Client-Open	81
Pedido de Configuração	54320
Recusa de Pedido acesso	733
Aceitação de pedido de core	1537
Aceitação de pedido de acesso	3857
Keep-Alive	140
FHO	17746

Tabela 1 - Tempos de resposta às solicitações do AR

As conclusões mais interessantes retiradas da análise destes resultados são as seguintes:

O tempo de aceitação de um novo cliente é extremamente rápido (81 μs) o que era esperado pelo facto de ser uma acção muito simples, só necessitando o QoS Broker de criar uma entrada na tabela dos AR ligados antes de poder responder;

O tempo resposta ao pedido de configuração é o mais elevado de todos os tempos que foram medidos, o que se deve ao facto de o QoS Broker necessitar de procurar qual o tipo de AR em questão e depois procurar na base de dados qual a configuração que lhe deve entregar. Esse valor não representa qualquer problema, uma vez que a configuração completa do AR só se faz quando este arranca;

A recusa de um pedido é mais rápida do que a aceitação porque quando o QoS Broker recusa um pedido não necessita reservar os recursos, uma vez que não há atribuição;

A resposta aos pedidos do core é mais rápida porque de acordo com a arquitectura da rede o core é uma zona "privilegiada" da rede de onde não se verificam os perfiz dos utilizadores;

O tempo de resposta a um pedido de *Fast Handover* é demasiado alto, podendo comprometer a resposta da rede ao movimento de um *Mobile Node*. Idealmente segundo os requisitos do projecto *Moby Dick* o tempo de resposta de toda a rede deveria ser da ordem do tempo que é actualmente gasto pelo QoS Broker. Existem dois motivos para que isto aconteça. Em primeiro lugar a implementação existente usa um vector para armazenar os dados dos ARs pertencentes à rede e a lista de sockets de comunicação com os ARs. Esse tipo de implementação torna lento o processo de procura e como no processo de FHO se começa por procurar qual o AR que tem o endereço indicado no pedido e depois se procura o socket que existe aberto para se poder enviar a resposta torna a resposta do QoS Broker algo lenta. Em segundo lugar o QoS Broker imprime no interface do utilizador dados referentes ao pedido recebido, aos serviços a transferir para o novo AR, bem como a composição do NVUP depois do processo de *Fast Handover* ter decorrido. A impressão dessa informação é feita de modo a dar informação ao administrador do serviços que são transferidos e como está a correr o processo de *Fast Handover*, mas por ser muito extensa adiciona muito tempo à resposta ao pedido de *Fast Handover*.

QoS Broker <=>AAAC

Foram também medidos valores dos tempos de resposta às solicitações do servidor de AAAC. A Tabela 2 mostra os valores medidos para cada um desses pedidos.

Mensagem	*Tempo de resposta (μs)*
Client-Open	84
Definição dos perfis de QoS	384305
Autorização de NVUP	4007

Tabela 2 - Tempos de resposta às solicitações do servidor AAAC

Analisando os valores da tabela pode-se concluir que: a resposta à mensagem de estabelecimento de ligação tem um valor bastante baixo (84 μs) de acordo com o que acontecia entre o AR e QoS Broker; a mensagem de definição dos perfiz de QoS contém muita informação que tem que ser lida, interpretada e armazenada pelo QoS Broker o que justifica que tenha uma resposta tão demorada. Não se trata de um problema de implementação, uma vez que esta acção só é executada quando o servidor de AAAC se liga; o tempo de resposta ao envio de um NVUP (*Network View of the User Profile)* é de 4007 μs, o que parece perfeitamente aceitável

QoS Broker <=>Radio Gateway

Como se pode verificar na Tabela 3 entre o QoS Broker e a RG só existe um tipo de mensagem. Entre a Radio Gateway e o QoS Broker existe sempre um AR que intervém no processo de pedido de recursos, que despoleta o pedido ao QoS Broker. Esse AR necessita ainda de uma resposta do QoS Broker de modo a que possa conduzir a informação que vem ou vai de ou para a rede da Radio Gateway. É essa a razão pela qual foram medidos dois tempos que aparecem na Tabela 3.

Mensagem	*Tempo de resposta em μs*
Contacto da RG	1079
Resposta ao AR	2388

Tabela 3 - Tempos de resposta a um pedido de um Mobile Node pertencente a uma rede de uma Radio Gateway

Verifica-se que o tempo de resposta à Radio Gateway é mais baixo do que ao AR, o que se deve à forma como o QoS Broker está implementado. Assim que o QoS Broker executa a rotina que faz o controlo de admissão e que decide aceitar um fluxo o QoS Broker executa logo a rotina que procura se a origem ou destino do fluxo a ser admitido está numa rede de uma Radio Gateway. Caso isso aconteça imediatamente ele envia a mensagem à Radio Gateway e só depois responde ao AR.

Verifica-se ainda que o tempo de resposta ao AR é mais elevado do que na situação em que não intervém a Radio Gateway. A explicação para esta diferença deve-se ao facto de que neste caso o QoS Broker necessita de perder algum tempo a construir e a enviar a mensagem à Radio Gateway.

QoS Manager

Atrasos ponto a ponto

Para medir este tipo de atraso, efectuou-se o registo de um *Mobile Node*, neste registo o *Mobile Node* efectua o registo de Mobile IPv6 no Home Agent, este pacote é então autorizado no QoS Manager, e o pacote de confirmação do Home Agent também é autorizado. Com a diferença de tempos entre a altura que o pacote parte do Mobile Node e chega a resposta pode-se calcular o atraso ponto a ponto do primeiro pacote (e resposta) enviado.

O atraso medido foi de 3,7 ms o que é bastante satisfatório.

Atraso em pedidos de serviço / autorização

Para a medição deste atraso pretendia-se medir o tempo entre a recepção do pacote que despoleta o pedido de serviço/autorização e o encaminhamento desse pacote. No entanto, esta medição não é de todo fiável devido à forma como o AR está implementado. Pelo que foi possível perceber, quando um pacote chega ao AR e necessita de ser autorizado, o AR descarta o pacote e envia o pedido de serviço/autorização ao QoS Broker e por consequência, caso a resposta seja positiva, a implementação do filtro. Entretanto o pacote descartado é substituído por um outro enviado pelo emissor do primeiro, como é possível perceber, este valor varia de acordo com a implementação do protocolo e da forma como o software usado foi construído.

De notar, que este processo ocorre tanto para acessos de core como para acessos da rede de acesso.

Sendo assim, não é possível efectuar as medições pretendidas.

Atraso na implementação ou actualização do filtro

Neste teste mediu-se o tempo entre a recepção do pacote COPS DEC que informa o AR que pode aceitar o fluxo em causa, e o envio do pacote COPS RPT que informa o QoS Broker que já configurou o filtro para esse fluxo.

Este atraso é de cerca de 416 μs.

Perda de pacotes

Devido à forma como o Access Router está implementado existe sempre perda de pacotes. Pois este descarta o pacote que despoleta o pedido de seerviço/autorização, como explicado anteriormente.

Desta forma é natural que haja uma perda de pacotes significativa caso os fluxos que os originam necessitem de ser autorizados. Caso haja uma demora na actualização/autorização de um fluxo os pacotes são perdidos pois não são retidos para que posteriormente possam ser encaminhados.

Paging

Nestes testes foram usados as versões *Dummy* de todo o *software* da rede *Moby Dick*, devido a ainda existirem problemas de integração do *software* de *paging* com os outros componentes.

Para o cálculo de todos estes atrasos, foi efectuado o mesmo teste 3 vezes, de modo a termos tempos fiáveis, e foram contabilizados os tempos de cada pacote. Os tempos apresentados são as médias dos 3 testes efectuados. Desta forma foi possível comprovar que a especificação dos diversos protocolos estava correcta através dos diagramas de fluxo de mensagens, para cada teste, retirados das capturas do *ethereal*.

Tempo total numa operação de paging

Pretende-se medir o tempo que decorre desde que se faz um pedido ao Mobile Node até que ele responde. Foi usado o comando *ping* e foi medido o tempo que demora a chegar um pacote do tipo *echo reply* ao Correspondent Node. Este tempo de atraso inclui o armazenamento de pacotes iniciais no Paging Agent, o processo de *paging*, o tempo de re-registo do Mobile Node na rede e finalmente o envio da resposta. Actualmente este valor cobre não só o tempo adicional do *paging* mas todo o processo até à resposta. De modo a obter uma comparação, pode-se medir o tempo de resposta de um Mobile Node activo, embora possa implicar um caminho diferente. Pretende-se também medir se existe ou não perda de pacotes. A azul encontram-se o *Ping Request* (1) e o *Ping Reply* (12), a laranja as mensagens do processo de *paging* (2,3,4,7,8,9,10,11) e a verde as mensagens de *Mobile IPv6* (5,6).

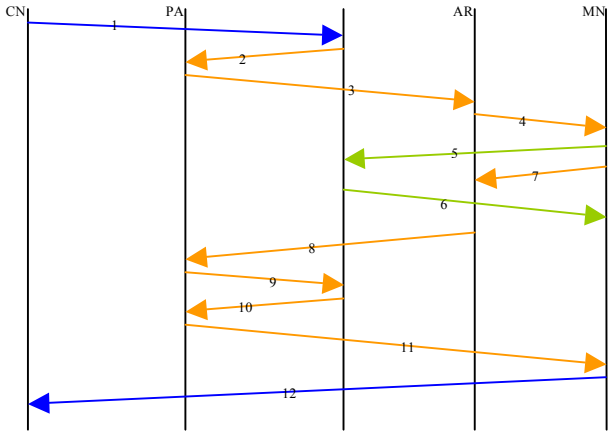

Diagrama 1 – Fluxo de mensagens trocadas numa operação paging

Nº	Origem (µs)	Destino (µs)	Diferença (µs)
1	0	147	147
2	238	579	341
3	1592	2246	654
4	2534	3791	1257
5	3995	6004	2009
6	6178	7924	1746
7	4280	7086	2806
8	7398	7711	313
9	8514	8878	364
10	8930	9408	478
11	9556	10962	1406
12	11056	13354	2298

Tabela 4 - Tabela de tempos do fluxo do Diagrama 1

O Diagrama 1 representado mostra o fluxo de mensagens trocadas entre os diversos componentes de rede para o teste em causa. A forma como o teste foi efectuado permitiu que com um único teste fosse possível efectuar as medições necessárias para outros testes planeados.

Pela tabela de tempos (Tabela 4) pode-se verificar que o tempo total de uma operação de *paging* é cerca de 13,4 ms. Este valor mostra um excelente desempenho da rede, tendo em conta que o maior tempo é gasto na propagação dos pacotes, principalmente quando é utilizada tecnologia *wireless*. Quanto a perda de pacotes, não se registou em nenhum teste de *paging* qualquer perda de pacotes.

Fast Handover

O objectivo deste teste é verificar a performance de um *Fast Handover* numa situação real, em que todo o software é usado e existe algum tráfego, de modo a simular uma utilização real.

Este processo tem em conta também as mensagens de AAAC que não influenciam a decisão de existir ou não *Fast Handover*, pelo que é possível concluir que todo o processo de *Fast Handover* termina com a recepção da mensagem 9 ao *Mobile Node*. No Diagrama 2 os diferentes protocolos estão identificados a cores diferentes, a azul estão a mensagens de *Fast Handover* (1,2,4,5,8,9), a verde as mensagens de QoS (3,6,7) e a laranja as mensagens de AAAC (10,11).

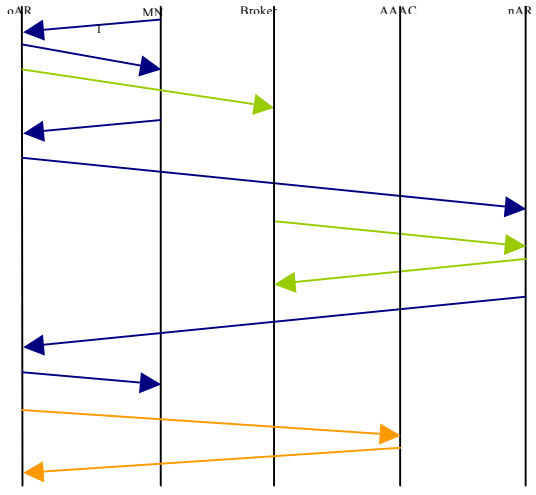

Diagrama 2 - Mensagens de Fast Handover com AAAC e QoS

Nº	Origem (µs)	Destino (µs)	Diferença (µs)
1	0	1657	1657
2	1938	3241	1303

3	2612	2844	232
4	3307	4749	1442
5	5677	6280	603
6	21366	21725	359
7	22898	23990	1092
8	23133	24294	1161
9	24568	25893	1325
10	122646	123208	562
11	139620	139966	346

Tabela 5 – Tempos medidos do fluxo do Diagrama 2

Latência Geral Durante o Fast Handover

Verificou-se que o processo de Fast Handover dura cerca de 140 ms. Tal como previsto o tempo necessário para efectuar o *Fast Handover* em si, sem ter em conta as mensagens de AAAC que não influenciam a decisão de *Fast Handover*, é o mesmo tendo todo o software integrado do que usando o *dummy* de AAAC, ou seja, cerca de 25 ms. Esta situação mostra a robustez de todo o processo e a atraso principal no conjunto integral encontra-se no tempo de espera entre as mensagens finais de *Fast Handover* e a primeira mensagens de AAAC a ser enviada.

Pacotes perdidos durante o Fast Handover

Nos testes de *Fast Handover* não foi registado nenhum pacote de sinalização perdido. No entanto, verificou-se que haviam pacotes, neste caso Ping, que eram perdidos durante um *Fast Handover*, este tipo de situação apenas acontecia quando se usava o software de QoS real. Deve-se portanto, a alguns problemas ainda existentes nesta componente que foram explicados anteriormente. Uma das razões é o facto de os pacotes de registo de *Mobile IPv6* não chegarem ao *Home Agent*, o que faz com que o *Mobile Node* não receba os pacotes que para ele são direccionados.

Atraso específico entre as QoS Manager e QoS Broker

As comunicações de QoS durante o processo de *Fast* Handover atrasam o referido processo em média 18 ms. Tal como referido anteriormente é justificável devido as verificações internas que o *QoS Broker* tem que efectuar de modo a garantir ao *Mobile Node* os serviços por ele contratados.

Atraso específico entre AAAC Attendant e AAAC Server

O atraso nas comunicações de AAAC em toda a rede durante o *Fast Handover* deve-se ao facto do oAR informar o AAAC Server dos valores contados durante o uso do *Mobile Node* desse AR. Pelo facto de estas mensagens terem um carácter informativo e não decisivo para a concretização ou não do *Fast Handover*, pode-se admitir que não existe atraso do ponto de vista do *Fast Handover* devido as comunicações AAAC.

Largura de banda de um fluxo TCP n um FHO

Verificou-se que a largura de banda na altura do Fast Handover (seg, 99023) decresce ligeiramente. Este decréscimo deve-se ao facto de as mensagens de *bicast* diminuir a largura de banda útil durante um processo de Fast Handover.

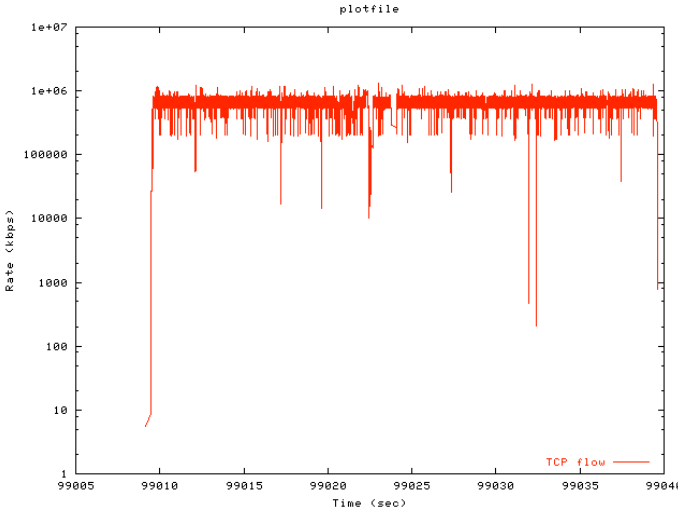

Figura 3 - Gráfico da largura de banda utilizada num FHO

VIII. SOFTWARE DESENVOLVIDO

PTQoSM

Tendo por base o QoSManager desenvolvido no âmbito do projecto Moby Dick e as suas limitações foi requerido que se desenvolvêsse um substituto capaz de substituir o QoSManager Moby Dick sem qualquer outra modificação ao restante software existente na rede Moby Dick. O suporte funcional deste software seria apenas QoS e não necessitaria de suportar FHO.

Tendo por base estes requisitos foi desenvolvido recorrendo às mesmas API's utilizadas pelo QoSManager do projecto Moby Dick um novo QoS Manager ao qual se chamou PTQoSM.

A principal lacuna que o PTQoSM combateu foi a flexibilização do numero de interfaces suportadas tanto a nível de core como a nível de acesso. Através de um desenvolvimento Orientado por Objectos em C++, criou-se uma abstração que permite por configuração ter várias interfaces de rede tanto ligadas à rede de core como à rede de acesso. Outro aspecto importante foi a substituição dos vários processos existentes no QoSManager Moby Dick por um só processo com Threads. Esta substituição torna não só o QoSManager mais estanque como substituiu a

necessidade de recorrer a memoria partilhada para troca de informação (processo lento para a aplicação em causa).

O uso das API's utilizadas pelo QoSManager Moby Dick restringiu as possíveis funcionalidades e capacidades do PTQoSM desde muito cedo. Nomeadamente o facto da COPS API não ser Thread safe.

Esta e outras limitações levaram a que se pensasse numa evolução de grau superior à do PTQoSM que veio dar origem ao SPAAQE, o qual se descreve de seguida.

SPAAQE

O SPAAQE é na sua essência um Edge Router de arquitectura DiffServ (DS), no entanto apresenta várias variações, que o tornam numa ferramenta poderosa de gestão e configuração de fluxos DS.

Assim sendo o SPAAQE poderá ser utilizado em redes em que é necessário fazer admissão de fluxos vindos de diversos utilizadores com requisitos muito personalizados.

O SPAAQE tem pois por audiência alvo Service Providers, que desejem dar aos seus clientes níveis de qualidade de serviço altamente configuráveis, mantendo um elevado grau de gestão sobre a sua rede.

O SPAAQE autentica, classifica e autoriza os fluxos por meio de informação trocada com uma rede QoS Broker's central[1]. Desta forma o SPAAQE torna-se ideal para redes que desejem ter vários Edge Routers com todas as funcionalidades do SPAAQE, e que necessitem constantemente de reconfigurações e/ou pequenas alterações.

Outra função importante do SPAAQE é a sua capacidade de processar os pacotes que o atravessam. O exemplo de um cenário onde esta função poderá ser útil, é na criação de uma directoria de endereços para serviços. Concretizando no caso do DNS, o SPAAQE poderia fazer correspondência entre DNS server, configurado no utilizador e o endereço de DNS que conste numa directoria configurada no SPAAQE. Desta forma o Service Provider poderia distribuir configurações aos seus utilizadores que nunca correriam risco de se tornar obsoletas, ou quando necessitasse de distribuir serviços por várias máquinas, não necessitaria de informar o utilizador do novo endereço do serviço.

A capacidade de sinalização de fluxos a terceiras entidades é uma opção flexível, que permite a expansão dinâmica de funcionalidades do SPAAQE.

Desta forma e através de todas estas novas funcionalidades o SPAAQE apresenta-se como uma alternativa consistente e de valor acrescentado ao QoS Manager do Projecto Mobydick.

COPSpp

Apesar de existirem já diversas implementações do protocolo COPS tais como API da Intel e a API utilizada no seio do Projecto Moby Dick, nenhuma destas responde aos requisitos estabelecidos para utilização no SPAAQE. A primeira por apenas disponibilizar source code para PEPs e o mesmo ser em C, a segunda por ser uma implementação demasiado simplista e também ela ser em C. Assim desenvolveu-se uma nova API Open Source em C++ que respeita-se o RFC 2748 (The COPS Protocol), fosse thread safe e implementa-se tanto uma interface para PEP's como para PDP's.

A COPSpp (leia-se COPS plus plus) é pois uma API C++ distribuída na forma de shared library que permite o desenvolvimento de aplicações com suporte a funcionalidades de comunicação via COPS.

IX. CONCLUSÕES

Quando se deu inicio à realização deste projecto os conceitos AAAC, QoS, FHO ainda não estavam bem interiorizados. No entanto, rapidamente a complexidade destes conceitos foram compreendidos, assim como da sua importância nas redes de comunicações baseadas no protocolo IP. O seu estudo não só alargou o espectro de conhecimentos na matéria como permitiu melhor compreender o que de novo se faz, nomeadamente no âmbito do projecto Moby Dick, com o qual o projecto esteve intrinsecamente ligado através das colaborações prestadas.

A participação no projecto Moby Dick foi ainda altamente enriquecedora na medida que permitiu interagir com as pessoas envolvidas num projecto de elevada complexidade, muitiequipa e multinacional. Esta experiência é duplamente enriquecedora em termos de conhecimentos adquiridos e experiência de trabalho em equipe. Com a colaboração prestada ao nível da definição de ambientes de teste, instalação de sistemas, debug de problemas e validação de soluções, este projecto foi ainda um elemento útil ao projecto Moby Dick, como fonte de feedback na comunidade para o trabalho desenvolvido no seio do projecto.

Através da projecção, instalação, configuração e manutenção da rede testes, adquiriu-se experiência na área da gestão de redes em especial utilizando o protocolo IPv6. A nível da configuração individual das máquinas Linux adquiriu-se experiência na sua administração pela escrita e modificação de scripts de boot e configuração das distribuições instaladas, o que contribuiu para o melhor conhecimento desta plataforma tão importante nos dias de hoje.

O desenvolvimento de software a nível do projecto, na figura do PTQoSM e SPAAQE deu ainda uma visão mais completa da complexidade inerente ao desenvolvimento de ferramentas para plataforma IP. Permitiu também consolidar conhecimentos na área de Eng. Software, Linguagens Orientadas a Objectos e Analise de Sistemas.

A presença do projecto nas conferências cientificas Wireless Summit 2003 e ConfTele 2003 permitiram ainda dar uma visão ao exterior do trabalho realizado a nível de projecto final de 5º ano, como também (e razão principal) do estado actual de desenvolvimento do Projecto Moby Dick. Foi uma experiência enriquecedora na medida que

se pode expor o trabalho e interagir com a comunidade recebendo feedback do mesmo.

Por fim o trabalho realizado, desenvolveu as capacidades dos seus elementos de proceder a trabalho em equipa, com troca de experiências e conhecimentos que valorizaram cada um dos elementos do grupo.

X. REFERÊNCIAS

[1] Victor Marques, Rui L. Aguiar, Carlos Garcia, Jose Ignacio Moreno, Christophe Beaujean, Eric Melin, Marco Liebsch, *An IP-based QoS architecture for 4G operator scenarios,* IEEE Wireless Communications Magazine (scheduled to be published in June 2003).

[2] Pedro Gonçalves, Diogo Gomes, Victor Marques, Rui L. Aguiar, QoS Control support for heterogeneous networks, CRC 2003, 2003.